#12
2/20/03
AW

PATENT
Attorney Docket No.: P3776
Client/Matter No.: 80168.0060
Express Mail No. EV215757070US

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of:

Paul W. Weschler, Jr.

Serial No.: 09/315,200

Filed: May 19, 1999

For: **MECHANISM AND METHOD FOR MANAGING SERVICE-SPECIFIED DATA IN A PROFILE SERVICE**

Art Unit: 2177

Examiner: K. PHAM

COMMISSIONER FOR PATENTS
WASHINGTON, D.C. 20231

APPELANT'S BRIEF UNDER 37 CFR 1.192

I. Real Party in Interest

Sun Microsystems, Inc.
4120 Network Circle
Santa Clara, CA 95054
USA

II. Related Appeals and Interferences

No other appeals or interferences are currently known to Appellants that will directly affect, be directly affected by, or have a bearing on the decision to be rendered by the Board of Patent Appeals and Interferences in the present Appeal.

III. Status of Claims

Pending claims 1-53 stand rejected under 35 U.S.C. § 102 as anticipated by U.S. Patent No. 5,838,970 to Thomas. On December 12, 2002, Appellants appealed

from the final rejection by filing a Notice of Appeal of all pending claims 1-53.

IV. Status of Amendments

No amendments were made to pending claims in response to the Final Office Action mailed on August 8, 2002. Therefore, claims 1-53, as previously amended, remain in the application for consideration in this appeal.

V. Summary of the Invention

Briefly stated, the present invention involves a profiling service (201, 202, 203) for accessing user data. The profiling service (201, 202, 203) includes a plurality of profile objects (500, 502) containing at least one true-data attribute (504). The true-data attribute (504) includes a true-data key and at least one true-data value field. At least one meta-data attribute (506, 508) is associated with the true-data attribute (504), and includes a meta-data key and at least one meta-data value field. Methods within each profile object (500, 502) access the user data according to the meta-data attribute (506, 508).

The true-data attribute (504) may include the user data, or an external reference to the user data. The profile object (500, 502) may include at least one true-data attribute binding to another one of the profile objects (500, 502). The meta-data key can be equated with the true-data key. Furthermore, the meta-data attribute (506, 508) can be identified with a prefix field in the meta-data value field. The methods to read and write the true and meta-data attributes may be included in the profile object. The methods may also set an owner, an access privilege, a group, a creation time, a update time, expiration time, a trigger location, a binding flag, and an assurance level of the true-data attribute. The profile service may include at least one profile level meta-data attribute.

The present invention is particularly useful in environments that require a data structure that is quickly searched and where the data is suited to a hierarchical representation. Also, the system of the preferred implementation is optimized to store and make available relatively compact units of data that serve to configure devices and computer environments rather than operational or analytical data upon which the

computer environment may operate at runtime. Hence, the present invention is best used when it stores and retrieves data that is frequently searched and retrieved, but infrequently changed.

To aid evaluation by the Board, definitions presented in the Specification are reproduced below:

Attribute - The combination of a key related to one or more values. Frequently described as a key=value pair.

Meta-data - Special data attributes used by a Profile Service to help maintain and manage Profiles and user data within the Profiles.

Binding - A special type of attribute where the value of the key=value pair is itself a Profile. Binding one Profile to another in a chained fashion it is possible build tree structures of related profiles and attribute information (called a profile tree).

Profile - A collection of attributes related either directly or indirectly to a EntityProfile that contains data or links to data used by an entity.

Profile Service - The term generally used to refer to all of the functional interfaces of Profile Service and/or an implementation of. The term is frequently used to refer specifically to the core Profile Service interface.

Profile Protocol - A data transport methodology that exposes functional interfaces in a language and platform independent manner. It is a distributed communication mechanism implemented in the examples herein as extensible markup language (XML) over hypertext transfer protocol (HTTP).

Profile Type - A special attribute of all Profiles. This attribute is mandatory for all profiles and plays an important role in generating resource IDs and data indexing.

VI. Issues

The sole issue is: Does the rejection establish a case of anticipation of claims 1-53 based on the Thomas patent?

VII. Grouping of Claims

Claims 1-53 stand or fall together.

VIII. Argument

The Rejection under 35 U.S.C. 102 is Deficient because the Thomas reference fails to show all of the elements of Independent claims 1, 18, 41, 42, 48 and 49.

Independent claims 1 calls for, among other things, acts of 1) associating at least one meta-data attribute with a true-data attribute, 2) storing the associated meta-data attribute in the profile object, and 3) managing the true-data attribute according to the associated meta-data attribute.

Independent claim 18 calls for, among other things, 1) at least one meta-data attribute that is associated with a true-data attribute, 2) a profile object containing both the true-data attribute and the associated meta-data attribute, and 3) methods within the profile object to access the true-data attribute according to the associated meta-data attribute.

Independent claim 41 calls for, among other things, 1) means for associating at least one meta-data attribute with a true-data attribute, 2) means for storing the associated meta-data attribute in the profile object, and 3) means for managing the true-data attribute according to the associated meta-data attribute.

Independent claim 42 calls for a profile object having 1) at least one true-data attribute and 2) at least one meta-data attribute that describes the associated true-data attribute.

Independent claims 48 and 49 each call for computer programs products that include computer code devices to implement the method of claim 1.

The enumerated elements above are neither expressly nor inherently described in the *Thomas* reference. Specifically, Thomas fails to show a method, system, data structure, or computer program product that

- associates at least one meta-data attribute with a true-data attribute;
- stores the associated meta-data attribute in the profile object; and/or
- manages the true-data attribute according to the associated meta-data attribute.

In the present invention true-data attributes are associated with meta-data attributes as part of a solution to a long-standing problem of locating up-to-date resources (e.g., user data) that are geographically and topologically scattered among components of a distributed computing system. *See Specification*, page 6, lines 28–29. Locating these resources presents a formidable challenge in distributed systems that rapidly add new software, equipment and users to the computing environment. *See Specification*, page 6, lines 11–27.

One previous solution to finding up-to-date resources on rapidly changing distributed systems was to use directories that centralize information about the system such as mail address books, printer locations, and public key infrastructure, among other types of information. *See id* at lines 29–32. Unfortunately, these directories often did not interact with each other and therefore developed redundant and inconsistent information when implemented on a rapidly changing distributed system. *See Specification*, page 6 line 35 to page 7 line 3.

In order to improve the speed and efficiency with which directories are updated to reflect changes in a distributed system, meta-directories were introduced that attempted unify and centrally manage the disparate directories within the system. *See Specification,* page 7, lines 3–5. Unfortunately, meta-directories handle only a finite range of predetermined data types that cannot be extended. *See Specification*, page 7, lines 25–27. New types of software and hardware on the system that were not foreseen by the meta-directory developer are difficult to manage with conventional meta-directories. *See id.*

The present invention overcomes the problems with directories and meta-directories on distributed systems by associating meta-data attributes with the true-data attributes, where the meta-data attributes can be used to manage the true-data

attributes. *See Specification*, page 8, lines 12–25. The associated true-data and meta-data may be located in the same object profile. *See id.* In this manner, one can determine how a true-data item is to be managed (e.g., who can access the true-data item...when was it created...who created it...can it be copied?...can it be changed?...can it be printed?...etc.) immediately upon initial access of the true-data item by examining the associated meta-data attribute(s).

In contrast, prior systems such as exemplified by *Thomas* required that one access a separate data structure to determine this management information about a true-data item. *Thomas*, for example, describes conventional directories (called "repositories" in the reference) for organizing information about a distributed system similar to the directories described above. *See, e.g., Thomas*, Abstract, lines 1–3 ("An object-oriented computer environment is managed by storing, in a plurality of repositories accessible during the life cycles of objects, information required to initiate object operations."). The repositories in *Thomas*, like other conventional directories, have to be upgraded by a system administrator. *See id*, column 4, lines 30–37 ("[T]he present development allows software upgrades to be made by an administrator merely changing the contents of appropriate information repositories.").

Thomas never distinguishes or uses the terms "true-data" and "meta-data". Hence, logic would dictate that the data repositories in *Thomas* include either "true-data" or "meta-data", but not both. However, the Examiner characterize Table 4 in the *Thomas* reference as showing a repository with attributes of both true-data in the form of "object types" and meta-data in the form of "object-type entries." *See* Office Action, pages 8–9, section 3. The Examiner further cites a description of Table 4 in the reference as support for the proposition that the meta-data attributes manage the true-date attributes in the repository. *See id.* The Appellant disagrees with this characterization of Table 4 in *Thomas*.

Table 4 contains "object-type entries" that "contain search parameters and implementation information for various object types." *See Thomas*, column 14, lines 28-35. *Thomas* states that the object-type search parameters contained in the object-type entries "permit the actual executables and libraries used to implement an object operation to be selected based on system needs or user preferences." *See id* (emphasis

added). Object-type search parameters, the alleged meta-data, are not used to manage other data in Table 4, but instead are used to manage "actual executables and libraries" outside the Table 4. Thus, based on the description of these parameters in the reference itself, there is no evidence that meta-data attributes in a table are being used to manage a true-data attribute in that same table.

In contrast with the presently claimed invention, the "search parameters" and "executable information" in Fig. 4 are both used to customize the instantiation of a remote object. Hence, if we are to characterize the "search parameters" and "executable information" as meta-data, that meta-data is not used to manage anything in the repository itself, but instead to manage some other object that may or may not even be executed on the same computer. This feature of Thomas actually teaches away from the present invention in that Thomas intentionally, explicitly disassociates the "search parameters" and "executable information" from the managed object by placing this meta-data into a separate information repository. Hence, one must access this separate repository before an object instance is created, and whether the correct information is used for a particular object instance will be controlled by the ability to synchronize the various information repositories—precisely one of the problems the instant invention is intended to solve.

The distinction between meta-data and true-data may be conceptually difficult, but is important to understanding of the present invention. By way of a specific example, analysis of Table 1 in the *Thomas* reference is useful. If we break Table 1 down for analysis, the entry labeled "Object#1" is a header having a value that identifies a name or unique ID for a particular object. The next entry, labeled "Location#1", indicates a place where object#1 is stored. Hence, Location#1 is a true-data attribute of object#1. The value contained in Location#1 is not meta-data because it does not describe anything about the name of "object#1". What is missing from Thomas is any indication of meta-data that would describe something about the true-data contents of each entry. For example, if Table 1 included an entry that indicated the name "object#1" contains 18 characters, or the name of "object#1" was accessible only to certain users, such an entry would be meta-data. Instead, each entry includes only an attribute of the object itself, not meta-data describing the true-

value of the attribute.

As noted above, claims 1, 18, 41, 42, 48, and 49 include as an element an act or a means for associating a true-data attribute according to an associated meta-data attribute, where the true-data attribute and meta-data attribute are in the same object profile. These elements are neither expressly nor inherently described in *Thomas*. Nor is this element suggested by the Thomas reference. Claims 2-17, 19-40, 43-47, and 50–53 depend from claims 1, 18, 42 and 49 respectively, and are believed to be allowable for at least the reasons set out above. Moreover, each of the dependent claims is believed to be independently allowable because they call for specific meta-data values that are not shown or suggested in the relied on reference. Accordingly, Appellant requests that the rejection of the claims under 35 U.S.C. § 102(e) over *Thomas* be withdrawn.

IX. Conclusion

In view of all of the above claims 1-53 are believed to be allowable and the case in condition for allowance and it is respectfully requested that the Examiner's rejection be overturned.

Respectfully submitted,

Date: February 12, 2003

BY: Stuart Langley

Stuart T. Langley #33,940
HOGAN & HARTSON LLP
One Tabor Center
1200 17th Street, Suite 1500
Denver, Colorado 80202
Phone: (720) 406-5335
Fax: (720) 406-5301

X. APPENDIX OF CLAIMS ON APPEAL

1(Twice Amended). A method for managing a profile service, the method comprising:

storing at least one true-data attribute in a profile object, said true-data attribute includes a true- data key and at least one true-data value field;

associating at least one meta-data attribute with said true-data attribute, said meta-data attribute includes a meta-data key and at least one meta-data value field, wherein the meta-data value field describes the associated true-data attribute;

storing said associated meta-data attribute in said profile object; and

managing said true-data attribute according to said associated meta-data attribute.

2. The method as defined in claim 1, wherein the method further includes indicating in said meta-data attribute an access privilege of said true-data attribute.

3. The method as defined in claim 1, wherein the method further includes indicating in said meta-data attribute an owner of said true-data attribute.

4. The method as defined in claim 1, wherein the method further includes indicating in said meta-data attribute a group of said true-data attribute.

5. The method as defined in claim 1, wherein the method further includes indicating in said meta-data attribute a creation time of said true-data attribute.

6. The method as defined in claim 1, wherein the method further includes indicating in said meta-data attribute a update time of said true-data attribute.

7. The method as defined in claim 1, wherein the method further includes indicating in said meta-data attribute a expiration time of said true-data attribute.

8. The method as defined in claim 7, wherein the method further includes:

identifying said true-data attribute with said expiration time beyond a profile service time; and

deleting said identified true-data attribute from said profile object.

9. The method as defined in claim 1, wherein the method further includes indicating in said meta-data attribute at least one trigger location of said true-data attribute.

10. The method as defined in claim 1, wherein the method further includes indicating in said meta-data attribute a binding flag of said true-data attribute.

11. The method as defined in claim 1, wherein the method further includes indicating in said meta-data attribute an assurance level of said true-data attribute.

12. The method as defined in claim 1, wherein the method further includes identifying said meta-data attribute information by a prefix field in said meta-data value field.

13. The method as defined in claim 1, wherein the method further includes:

associating at least one profile level meta-data attribute to said profile object;

storing said profile level meta-data attribute; and

managing said profile object according to said associated profile meta-data attribute.

14. The method as defined in claim 13, wherein the method further includes indicating in said profile level meta-data attribute at least one template resource ID of said profile object.

15. The method as defined in claim 13, wherein the method further includes indicating in said profile level meta-data attribute an object class of said profile object.

16. The method as defined in claim 13, wherein the method further includes indicating in said profile level meta-data attribute an object ID of said profile object.

17. The method as defined in claim 13, wherein the method further includes indicating in said profile level meta-data attribute a binding resource ID of said profile object.

18(Twice Amended). A profiling service for accessing user data, said profiling service comprising:

a plurality of profile objects;

at least one true-data attribute contained in each of said profile objects, said true-data attribute includes a true-data key and at least one true-data value field;

at least one meta-data attribute associated to said true-data attribute and contained in said profile object, said meta-data attribute includes a meta-data key and at least one meta-data value field, wherein the meta-data value field describes the associated true-data attribute; and

methods within each profile object to access the user data according to said meta-data attribute.

19. The profile service of claim 18, wherein said true-data attribute comprises the user data.

20. The profile service of claim 18, wherein said true-data attribute comprises an external reference to the user data.

21. The profile service of claim 18, further comprising at least one true-data attribute binding to another one of said profile objects.

22. The profile service of claim 18, wherein said meta-data attribute is identified with a prefix field in said meta-data value field.

23. The profile service of claim 18, wherein said meta-data key is equated with said true-data key.

24. The profile service of claim 18, further comprising methods within said profile objects to read and write said true-data attribute.

25. The profile service of claim 18, further comprising methods within said profile objects to read and write said meta-data attribute.

26. The profile service of claim 18, further comprising methods within the profiling service to set an owner of said true-data attribute.

27. The profile service of claim 18, further comprising methods within said profile objects to set an access privilege of said true-data attribute.

28. The profile service of claim 18, further comprising methods within said profile objects to set a group of said true-data attribute.

29. The profile service of claim 18, further comprising methods within said profile objects to set a creation time of said true-data attribute.

30. The profile service of claim 18, further comprising methods within said profile objects to set a update time of said true-data attribute.

31. The profile service of claim 18, further comprising methods within said profile objects to change a expiration time of said true-data attribute.

32. The profile service of claim 31, further comprising methods within said profile objects to delete said true-data attribute with said expiration time beyond a profile service time.

33. The profile service of claim 18, further comprising methods within said profile objects to set a trigger location of said true-data attribute.

34. The profile service of claim 18, further comprising methods within said profile objects to set a binding flag of said true-data attribute.

35. The profile service of claim 18, further comprising methods within said profile objects to set an assurance level of said true-data attribute.

36. The profile service of claim 18, further comprising at least one profile level meta-data attribute.

37. The profile service of claim 36, further comprising methods within said profile objects to set a template resource ID of said profile object.

38. The profile service of claim 36, further comprising methods within said profile objects to set an object class of said profile object.

39. The profile service of claim 36, further comprising methods within said profile objects to set an object ID of said profile object.

40. The profile service of claim 36, further comprising methods within said profile objects to set a binding resource ID of said profile object.

41(Twice Amended). A profiling service for accessing user data, said profiling service comprising:

means for storing at least one true-data attribute in a profile object, said true-data attribute includes a true-data key and at least one true-data value field;

means for associating at least one meta-data attribute with said true-data attribute, said meta-data attribute includes a meta-data key and at least one meta data value field, wherein the meta-data value field describes the associated true-data attribute;

means for storing said associated meta-data attribute with said associated true-data attribute; and

means for managing said true-data attribute according to said associated meta-data attribute.

42(Twice Amended). A profile object for maintaining client configuration data in a hierarchical fashion, the profile object comprising:

at least one true-data attribute in the profile object, said true-data attribute includes a true-data key and at least one true-data value field; and

at least one meta-data attribute associated to with said true-data attribute and stored in said profile object, said meta-data attribute includes a meta-data key and at least one meta-data value field, wherein the meta-data value field describes the associated true-data attribute.

43. The profile object of claim 42, wherein said meta-data attribute specifies an access privilege of said true-data attribute.

44. The profile object of claim 42, wherein said meta-data attribute specifies a expiration time of said true-data attribute.

45. The profile object of claim 42, further including an aging method for deleting said true-data attribute with said expiration time beyond a profile service time.

46. The profile object of claim 42, wherein said meta-data attribute specifies a binding flag of said true-data attribute.

47. The profile object of claim 42, further including at least one profile level meta-data attribute associated to the profile object.

48. A computer program product comprising:

a computer usable medium and computer readable code embodied on said computer useable medium for causing the managing of a profile service, the computer readable code comprising:

computer readable program code configured to cause the computer to effect the storing of at least one true-data attribute in a profile object, said true-data attribute includes a true-data key and at least one true-data value field;

computer readable program code configured to cause the computer to effect the associating of at least one meta-data attribute with said true-data attribute, said meta-data attribute includes a meta-data key and at least one meta-data value field, wherein said meta-data key is equated with said true-data key;

computer readable program code configured to cause the computer to effect the storing of said associated meta-data attribute; and

computer readable program code configured to cause the computer to effect the managing of said true-data attribute according to said associated meta-data attribute.

49. A computer program product embodied in a tangible media comprising:

computer program devices readable by a data processor coupled to the tangible media for managing a plurality of profile data structures, each profile data structure comprising a hierarchical structure of true-data attributes and meta-data attributes, the computer program product comprising:

first program code devices configured to cause the data processor to store said true-data attributes in said profile data structures, said true-data attributes include a true-data key and at least one true-data value field;

second program code devices configured to cause the data processor to associate said meta-data attributes with said true-data attributes, said meta-data attributes include a meta-data key and at least one meta-data value field, wherein said meta-data key is equated with said true-data key;

third program code devices configured to cause the data processor to store said associated meta-data attribute; and

forth program code devices configured to cause the data processor to manage said true-data attributes according to said associated meta-data attributes.

50. The computer program product of claim 49 wherein the tangible media comprises a magnetic disk.

51. The computer program product of claim 49 wherein the tangible media comprises an optical disk.

52. The computer program product of claim 49 wherein the tangible media comprises a propagating signal.

53. The computer program product of claim 49 wherein the tangible media comprises a random access memory device.